[MISLAID PREV. COPY UP (S?)]

OSMIC
Preliminary Specifications

Note: this is a simplified version of the previous OSMIC document of May 96, which will still be available.

OSMIC

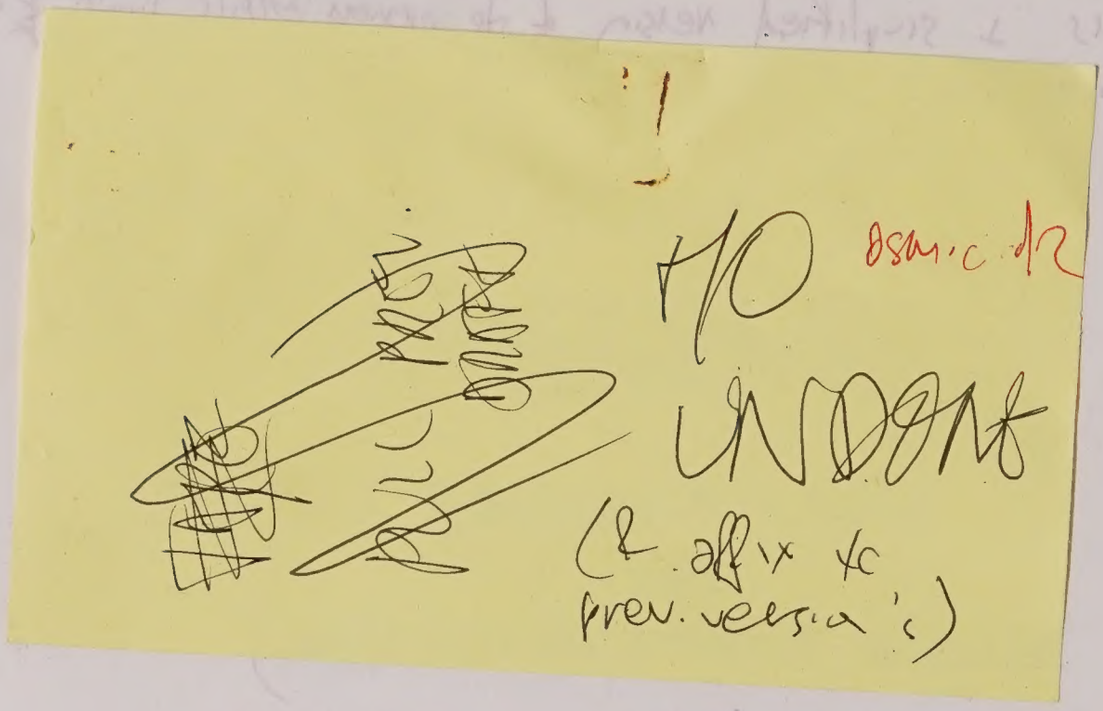
Open Standard for Media InterConnection

(Structure)

An Integrated Universal Format
especially for
TEXT, HYPERTEXT, ~~GRAPHICS~~ MULTIMEDIA,
VERSIONS, and INTERCOMP.

(Prof.) Theodor Holm Nelson
Keio University Shonan Fujisawa Campus
Kanagawa Prefecture, Japan
and
Project Xanadu, Sausalito, California

Permanent e-mail: ted@xanadu.net



osmic.shopnotes.d2
97.03.28
==========

OSMIC SHOPNOTES:
Notes for the OSMIC Workshop at Keio SFC,
Spring 1997.

(Prof.) Ted Nelson
Department of Environmental Information
e-mail ted@xanadu.net
Office Kappa 506
Office Hours: Wednesday 4-6

Preliminary Remarks

TODAY'S INSANE COMPUTER WORLD

Today's computer world is based on trapping users in "applications" which are controlled by individual manufacturers and get bigger and buggier all the time.

These applications typically have a warped idea of what users need, especially in the text areas.

Moreover, each application typically has ITS OWN methods of representing connections among data parts, and collections of data parts.

I believe we should have a common framework in which CONNECTIONS AND COLLECTIONS ARE ALL HANDLED IN THE SAME WAY, and anyone may produce visualizations and interface applets that work across a broad spectrum of materials.

THE HOPELESSNESS OF COMPUTER TEXT SYSTEMS

Today's text systems are a mess, practically worthless. "Word processing" simulates letter tiles on two-dimensional sheets, "desktop publishing" (programs for what used to be called *layout*) add font sugar to word processing, and "outline processing" uses clumsy methods for crude structure management and item control.

THE VITAL ISSUES OF VERSION MANAGEMENT AND ITEM CONTROL, OVERVIEW AND STRUCTURE MANAGEMENT

The main issues of text systems, I believe, are version management (keeping track of different versions and comparing them and knowing what's in them) and item management (keeping track of items to be included, and which items are at what locations in which versions). The only software on the personal market for item control is the "outline processor," which assumes:

1. That you are creating a sequential document (which you may not be).
2. That the best way to create a sequential document is to create a list of items in sequence,

then to change the sequence of that list of items continually. You cannot make different piles, for instance (as in the KJ editor).

The only software for version management is so-called "CASE" software (very expensive and clumsy), and certain utilities for Unix, which keep track only of differences between *lines* of text.

By structure management I mean keeping track of the main form of an object as it changes and changes in emphasis. This is the problem of overview, which overlaps with item control.

NO WAY TO MAKE NOTES ON THINGS

There is no way to make notes on your writings, comparable to writing in the margin or adding sticky notes.

We should even be able to make parallel columns of annotation throughout any objects, rather like spreadsheet columns, but which are only next to the parts we wish to comment on.

We need systems for such annotation, and they should have the same form as other links. Long-term and short-term annotations should have the same implementation.

HYPERTEXT

Editing Hypertext--

The popular forms of hypertext, especially World Wide Web and HyperCard (which is World Wide Web without the Internet), allow only drastic editorial changes that are hard to keep track of. Version management in HTML, in particular, is extremely difficult and annoying.

Hypertext Links--

HTML links
- break when the documents are edited
- can't be followed in both directions
- don't have types

All these things need to be corrected, but i do not believe they can be corrected within the framework of HTML.

The World Wide Web Consortium is working on all these problems. However, I want to proceed with what I think is the best solution and demonstrate it.

THE HOPELESSNESS OF COMPUTER FILING SYSTEMS

Computer filing is still all hierarchical, even though most projects overlap and often have many versions. (And even though categories in the real world often overlap and represent incorrect representations of the world.)

TO CORRECT THESE PROBLEMS--

It is to correct these problems that the Xanadu system, and the software we will be working on this semester, have been designed.

OSMIC

Preliminary Specifications

OSMIC is intended to be a rich, clean, simple compatibility standrd for interconnection and change among diverse software families; to improve the Web with enhanced relational and publication structures; to turn back the tide of baroque, proprietary data structures; and to allow all parties to create interface applets in any style with compatible results. Thus it is a natural complement to Java* and the Unix* philosopy.

It offers Mix-and-Match methods and need not be adopted as a whole to provide benefits. It continues the proud Xanadu tradition on a public-domain basis:

$$\frac{\textbf{OSMIC}}{\textbf{XANADU*}} = \frac{\textbf{POSIX}}{\textbf{UNIX*}}$$

OSMIC is a data structure. (OO is all very well, but it doesn't allow you to create *new visualizations*; OSMIC does.)

At present it is particularly intended for
- TEXT
- AUDIO SAMPLES
- MIDI ARRANGEMENTS
- VIDEO FRAMES
- PROGRAMS
- ANIMATIONS

and other highly sequential data in which separate elements may be addressed.

---

* trademarks of somebody.

B/B

OSMIC is intended to be a public-domain version of the Xanadu system, dealing with these matters as simply as possible. It is intended (to quote Marlene Mallicoat) as an INTEGRATED UNIVERSAL FORMAT" for text and other media.

In this workshop we will work on preliminary implementations of this structure for text, especially for e-mail, as examples of what will be possible with rich structure, rich linking and transclusion.

- This is essentially an IMPLEMENTATION-INDEPENDENT SPECIFICATION OF ALL THE XANADU STRUCTURES--
  - without surplus meaning
  - as public-domain structure
  - for general understanding
  - for general use
  - as an EXCHANGE STANDARD FOR COMPLEX OBJECTS
  - as a system for SHARABLE ANNOTATION (cf. Lotus Notes)
  - as a BASE STRUCTURE FOR EVOLVING DOCUMENTS ••• own tracking structure ••• not stuck with having to maintain own tracking system
  - for VERSION MANAGEMENT (successive)
  - for MAINTAINING INTERCONNECTION AMONG CHANGING DOCUMENTS

os.shn.adds.d2
[parallel to osmic.shopnotes.d2]
97.03.28
==========

LIMITS IN THE FIRST IMPLEMENTATION.

We will consider for the time being ONLY SEQUENTIAL TEXT WITH LINKS (like HTML), in order to make implementation simpler. (Other hyperstructures, which require a structure map, will not be in the first implementaiton.)

Links will not be editable in the first implementation.

Files will not be purgeable in the first implementation.

The system consists of:

DATA STRUCTURES:

1. FROZEN TEXT FILES (Append-and-Read-Only). Arriving text material is appended to the end and not moved again. Most operations deal with addresses of text segments in these frozen files.

2. FROZEN EDIT FILES (Append-and-Read-Only). An edit change which a user makes in a current version is assigned an edit-op index and appended to the edit-op files.

3. CURRENT VERSION LIST(S) of documents. A version is ONE LIST OF ADDRESSES pointing to the frozen text files and links. The addresses of the text segments are the succeessive pieces of text in the text file, followed by the links which are contained in the document.

The sequence of the version list goes: TEXT SEGMENTS IN ORDER; LINKS IN ORDER OF THEIR IDs. (Since a link can connect to multiple spans of material, it is not in a particular place in the document, as in HTML.)

4. LINKS. A link is TWO LISTS OF ADDRESSES pointing to the frozen text files or elsewhere. One list (the Left List) points to all the bytes on the left side of the link, the other list (the Right List) points to all the bytes on the right side of the link.

(5. LINK VERSIONS. If links are editable, the current version of a link is maintained like a document.)

(6. TRANSCLUSIONS are important, but they are not explicit in the data structure; a transclusion is the same element-addresses shared by two documents.)

Let's look at these ...]

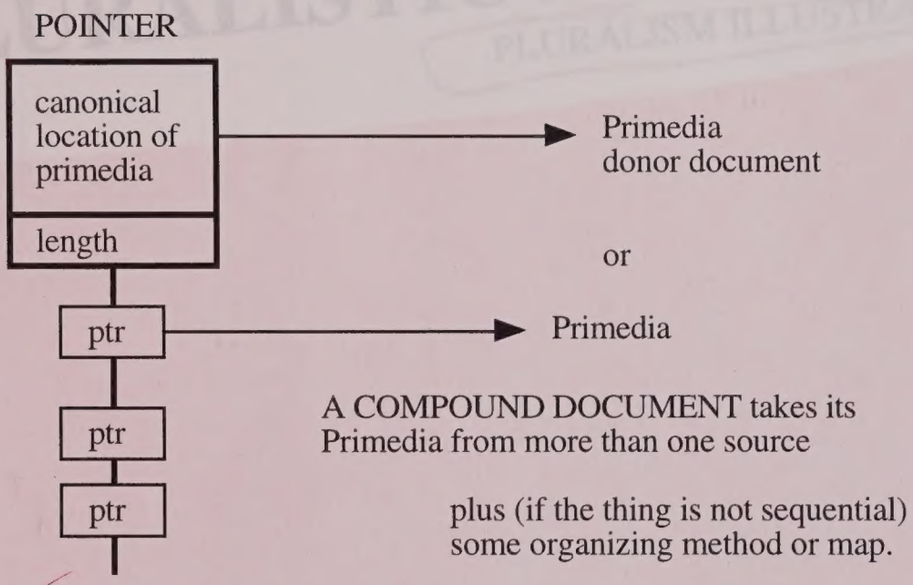
Basic concept:

the VERSION.

The operative unit is the *version*, not the document.

Relationships among versions are handled the same way, regardless of whether they are versions of the same thing.

A version is typically virtual, a list of pieces of frozen material (or "Primedia")



More formally, a version is

OSMIC
Preliminary Specifications

CONNECTIONS II: HIGHLY GENERALIZED PLURALISTIC LINKS.

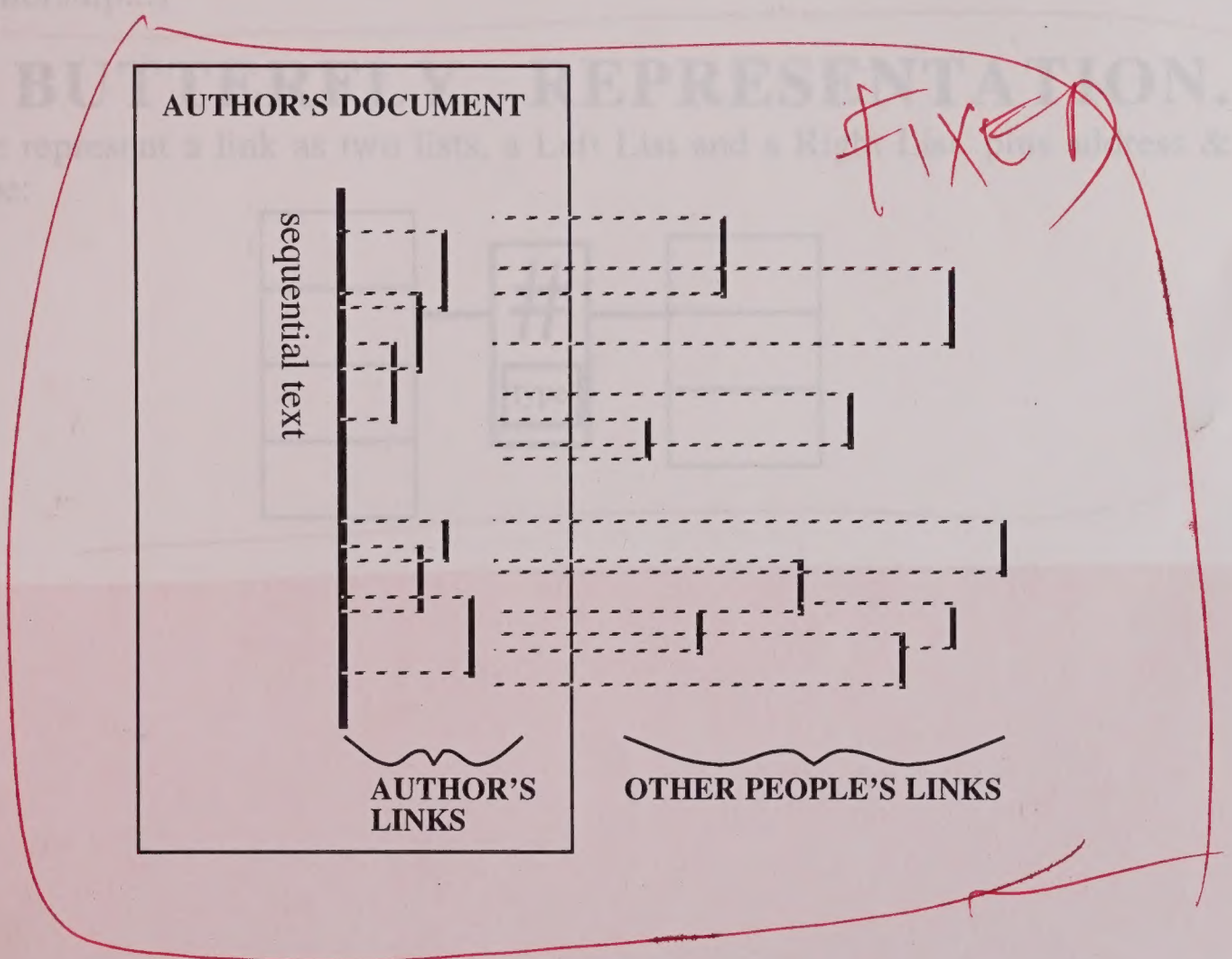
PLURALISM ILLUSTRATED, next page.

WHAT DO WE MEAN BY 'PLURALISTIC LINKS'?

We mean that everybody must be able to make first-rate links to anything.

WRONG APPROACHES: in HTML, only the author can create links. In some hypermedia packages, users are allowed to "create links" -- but they are decidedly inferior.

In the OSMIC standard, THE AUTHOR'S LINKS ARE NO DIFFERENT FROM ANYONE ELSE'S IN ¨IMPLEMENTATION (though superior in prestige and legitimacy.) For instance:

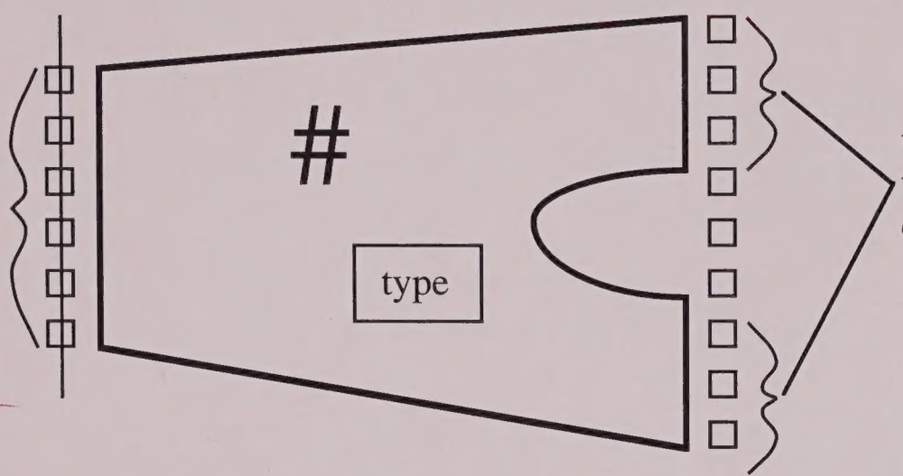


Note that all links are implemented identically and may be followed identically.

This is what we mean by Pluralism.

A link is between spans of elements, which may be scattered anywhere in the world. The Trouser Visualization:

The link is



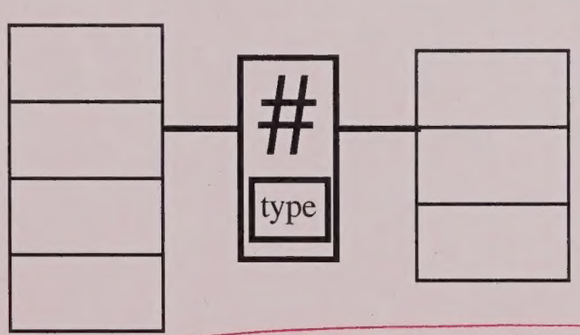
from these elements

to these elements.

It has a number (or address) and a type (coment, illustration, disagreement, authorship...)

BUTTERFLY REPRESENTATION.

We represent a link as two lists, a Left List and a Right List, plus address & type:



This, then, is the BUTTERFLY TROUSER LINK, a term which reinforces understanding.

LINK OPERATIONS are nontrivial, requiring many address comparisons. For instance, to follow a link from Version A to Version B:

Find addresses in Version A contained in link
Find addresses in Version B contained in link
Display as appropriate.

From these data structures it will be clear that in this system of representation, A TRANSCLUSION IS NOT A LINK. While some systems implement transclusion as a kind of link, that would create various heavy limitations.

CONNECTIONS I: TRANSCLUSIONS.

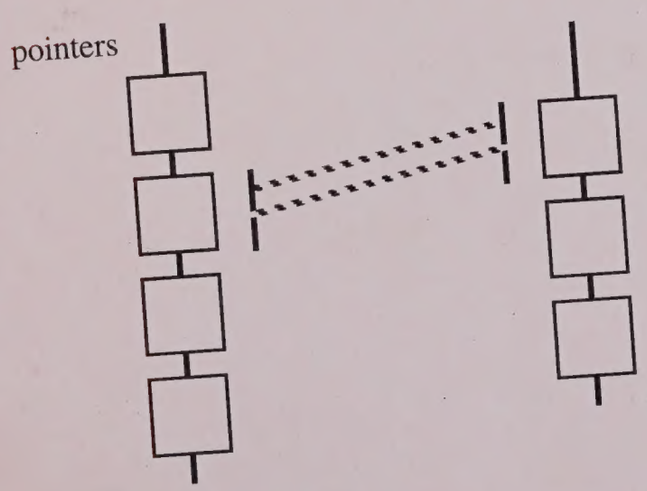
ABOUT TRANSCLUSION.

Always the heart of Xanadu, the transclusion principle means that *conceptually there is only one copy of anything*. Any piece of text, audio, etc., is treated as a *manifestation of a cosmic original*. (When you see a God or Saint in the road, is he a copy?) CRITERIA: Link to any portion is present on all manifestations; different contexts viewable side-by-side; feed and sale are from original.)

This means that the principal use of computer resources is to maintain the entanglements (in the quantum-physics sense) among the various instances.

Think of transclusion as a particular form of *identic data relationship*. There are various forms of identic data relationship, different in key details: copy, instance, counted reference, write-through cached copy, write-back cached copy... Transclusion is a highly general type.

Transclusion is basically very simple:



OVERLAP BETWEEN POINTERS IN TWO VERSIONS signifies transclusion in the data structure.

HOW IDENTITY REMAINS NOTICED BY THE SYSTEM is an implementation problem.

Its benefits can include:

UNRESTRICTED QUOTATION with automatic royalty paid by each downloader to each original publisher (the famous Xanadu payment model)

transclusive COMPARISONS OF CONTEXTS, supporting DEEP INTERCOMPARISON

transclusive EXCURSIONS (you go examining some other context awhile)

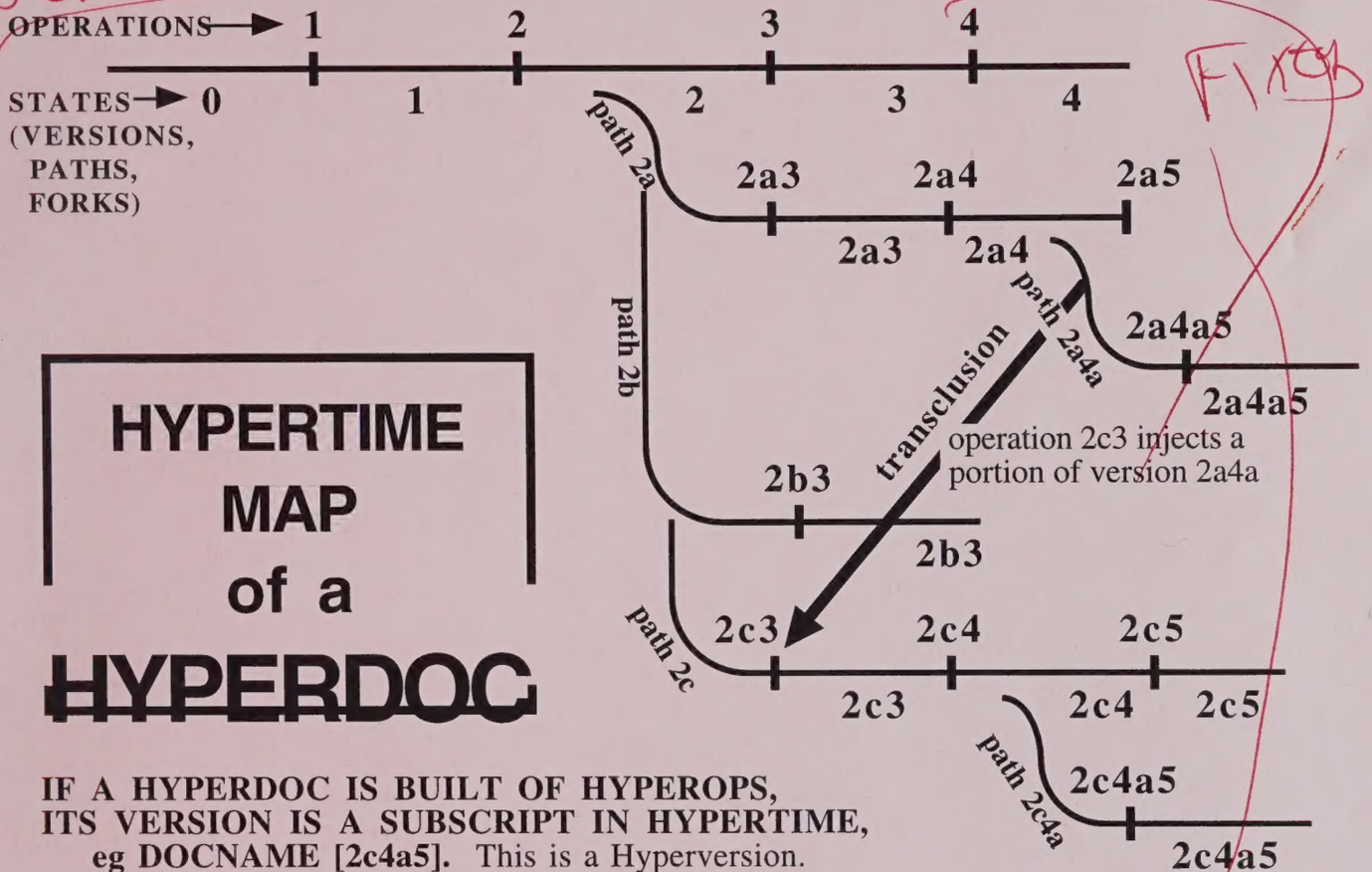
transclusive ROTATIONS (you decide you like another context better)

transclusive OVERVIEWS, OVERLAYS (pieces may be recomposited in views which lead back to the original contexts) etc.

OSMIC

Preliminary Specifications

HYPERTIME MAP of a HYPERDOC



**IF A HYPERDOC IS BUILT OF HYPEROPS,
ITS VERSION IS A SUBSCRIPT IN HYPERTIME,**
eg DOCNAME [2c4a5]. This is a Hyperversion.

THE REVERSIBLE HYPEROPS:

INSERT

Put new address pointers into the sequence.

REARRANGE

Change the address pointers.

(DELETE

is actually REARRANGE TO LIMBO. We suggest, as a practical hack, rearranging to negative addresses.)

TRANSCLUDE

(≡ "JOIN," compl. of version FORK)

INSERT POINTERS TO MATERIAL FROM ONE VERSION INTO ANOTHER.

MAKE/CHANGE LINK

MAKE/CHANGE STRUCTURE MAP

(Note that Support mechanisms for tracking or jumping to any particular Version-State are local and implementation-dependent.)

OPERATIONS: ==========

SHOW TEXT:
Show text version on screen.
Scroll on screen in version.

GENERAL FORM OF EDIT OPERATION:
Receive edit operation from user, assign it an index, append it to frozen edit files, and carry it out.

Example: INPUT TEXT:
Receive text from user, insert in displayed text, append to frozen text file, change version list

Example: REARRANGE TEXT:
Change the sequence of piece addresses in version list. (Create smaller pieces to rearrange if user specfies new cut-positions.)

LINK FOLLOW OPERATION:
Follow link L from version A to version B:
Find bytes in left list and right lists of L which are in A and B. (Make a temporary table.)
Highlight portions of L which are on screen in A.
Show portions of L which are in B. (User choice or programmer design required if set is large.)

FOLLOW-TRANSCLUSION OPERATION from version A to version B:
Find addresses which are the same in A and B.
Highlight them in A, show corresponding portions in B. (User choice or programmer design required if set is large.)

HO d2 ----------

OSMIC standard, intended as a stable description language for connected documents and media structures of all types. /96.10.10

"FILE SYNCHRONIZATION" A JOKE

INTRO FOR VIDEO ==========
(We don't know about the patent situation here)
Dreadful condition of software today.
Text systems are COMPLETELY UNRELATED to what serious writers and editors need
Want PARALLEL ANNOTATION
Want KNOWING RE-USE
Want a STRUCTURAL PARADIGM showing INTERCONNECTIONS AMONG MANY VERSIONS
• Want to see STRUCTURE OF VERSIONS AND CONNECTIONS
• Obj-oriented paradigm has gone in wrong direction •• NO CONTROL OF VIZN
• Want to CREATE NEW VIZNS NOT MADE AVAILABLE BY ORIGINATORS
• HTML, SGML are DORKY and go in the WRONG DIRECTIONS

• Visualize the user as an individual or company which is modifying and making variations of documents FOR DECADES, with many variants
• WANT MARGINAL NOTES BY ANYONE
• WANT LINKS BY ANYONE
• HTML BREAKS
• HTML LINKS VISIBLE IN ONLY ONE DIRECTION
• HTML DOES NOT SUPPORT LINKS ACROSS VERSIONING
• WANT TYPED LINKS (ranging from PUBLISHED to STICKY NOTES)

• For all this, a common structure has been developed by the Xanadu project over the years. Most of this was my own work in the sixties, but an important conceptual contribution was made by the Xanadu group in 1979-80.

OSMIC Premises ==========
Problems
• ITEM MANAGEMENT • VERSION MANAGEMENT
• KNOWING RE-USE for
•• Intercomparison
•• Coordination
•• Understanding of all uses of the particular material

FROZEN DATA =====

TRANSCLUSION =====

RICH LINKAGE =====

BRANCHING UNDO, MAINTENANCE OF INTERCOMPARABLE VERSIONS
=========

USES ==========
• Working-List management
• MAINTAINING DIFT VERSIONS
•• Authors
•• Law offices
•• Legislators
• VERSION COORDINATION

WE'LL TALK TEXT FIRST =========
• Text as the base data
• Sequential text as the virtual structures

DATA PARTS OF THE SYSTEM =========
• Append-and-Read-Only Files for Text

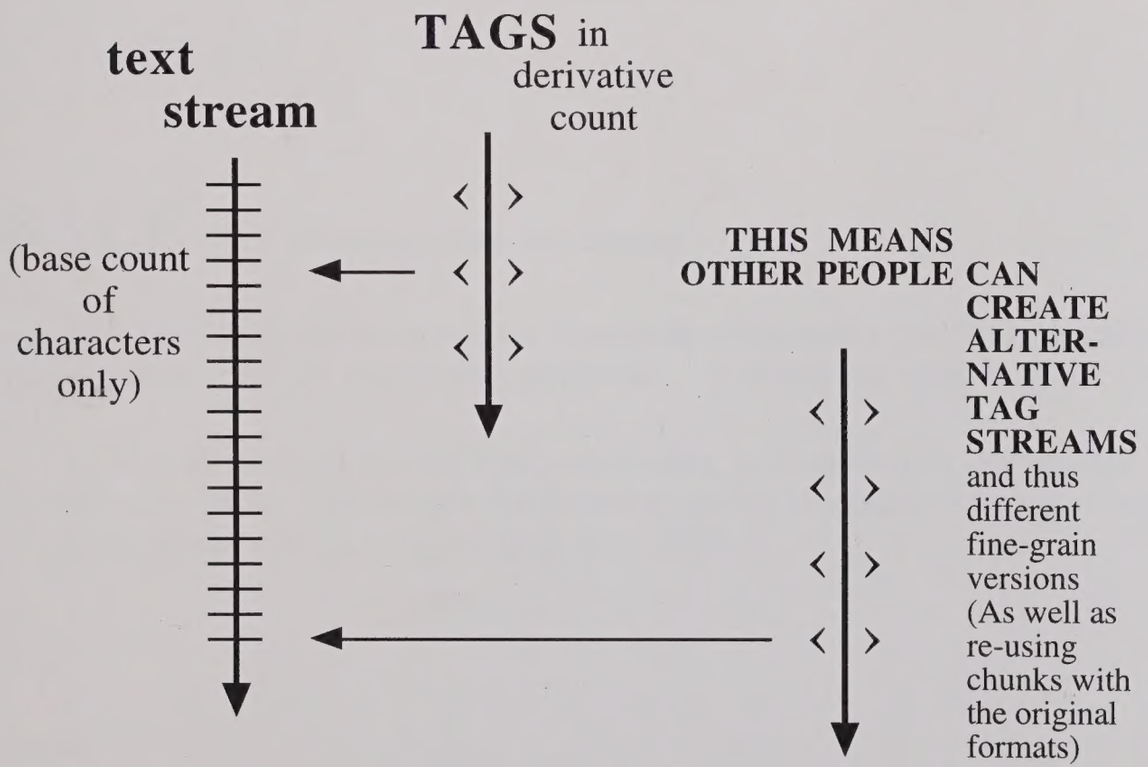
- **Append-and-Read-Only file for Edit Ops**
- **Virtual Sequential Document (list of base addresses**

Everything is assumed to be made of pieces with canonical network addresses. (How cached copies are resolved to their global addresses is a matter of local discretion.) These addresses point to the original contents, or Primedia.

Background recommendation:

PARALLELIZATION, NO EMBEDDED DATA. We recommend breaking out each form of data into parallel streams wherever possible, to facilitate addressing, re-use and pluralistic markup.

For instance, SGML or HTML can be broken out as:


text stream
TAGS in derivative count
(base count of characters only)
THIS MEANS OTHER PEOPLE CAN CREATE ALTER-NATIVE TAG STREAMS and thus different fine-grain versions (As well as re-using chunks with the original formats)

(By courtesy, SGML amd HTML may retain their native forms, but the

character count ignores the tags.)

OWNERSHIP

You may own Primedia Elements. (Which may be transcluded.)

You may own Links. (Which may be transcluded.)

You may own a Nameplex (which may be transcluded to another nameplex, provided that the receiving Nameplex is unique).

However, for SIMPLE, CLEAN IMPLEMENTABILITY, I suggest that only copies of Primedia be salable.

NOTE THAT BECAUSE ALL PRIMEDIA ELEMENTS ARE OWNED, TRANSCLUSION IS DEEMED TO BE FROM AN ORIGINAL HYPERDOC OR VERSION TO ALL ITS MANIFESTATIONS.

1.4- FLUSH

SALE, esp. of compound documents.

The user buys (downloads) a compound document purchasing each portion of primedia from its respective publisher. A receipt is sent with each portion.

(There is no cost for virtually including primedia in a document; only the downloader pays. (Although the quoting author is assumed to have bought one copy, or could not be constructing this work.))

OWNERSHIP

You may own Primedia Elements. (Which may be transcluded.)

You may own Links. (Which may be transcluded.)

You may own a Nameplex (which may be transcluded to another nameplex, provided that the receiving Nameplex is unique).

However, for SIMPLE, CLEAN IMPLEMENTABILITY, I suggest that only copies of Primedia be salable.

NOTE THAT BECAUSE ALL PRIMEDIA ELEMENTS ARE OWNED, TRANSCLUSION IS DEEMED TO BE FROM AN ORIGINAL HYPERDOC OR VERSION TO ALL ITS MANIFESTATIONS.

1.4- FLUSH

SALE, esp. of compound documents.

The user buys (downloads) a compound document purchasing each portion of primedia from its respective publisher. A receipt is sent with each portion.

(There is no cost for virtually including primedia in a document; only the downloader pays. (Although the quoting author is assumed to have bought one copy, or could not be constructing this work.))

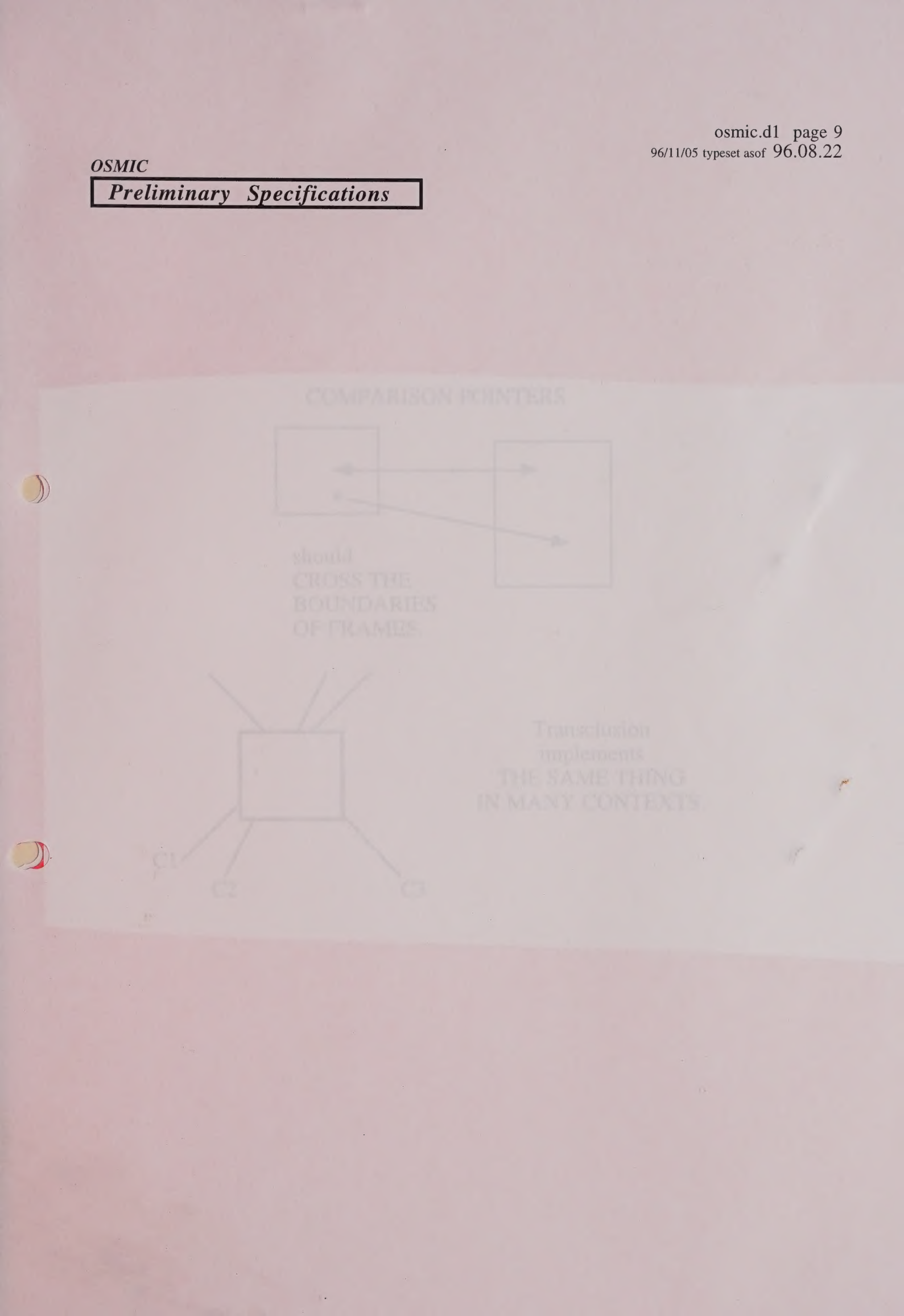
IF A HYPERDOC IS BUILT OF HYPEROPS, ITS VERSION IS A SUBSCRIPT IN HYPERTIME,
eg DOCNAME [2c4a5]. This is a Hyperversion.

The OSMIC
HYPERDOC is a highly generalized document, potentially with time travel and persistent links and transclusion across time travel.

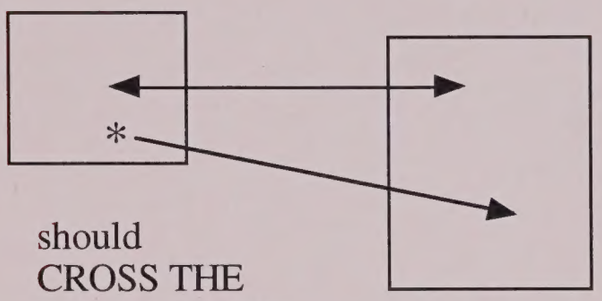
A Hyperdoc is:

A NAMEPLEX (universal ID mostly assigned by user or installation)
denoting
ONE OR MORE VERSIONS
and/or
HYPEROPS.
Hyperops are reversible operations mappable in hypertime.

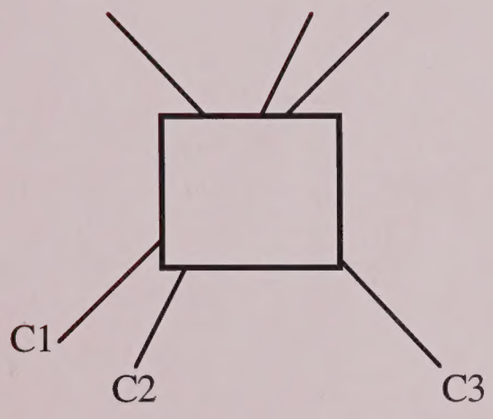
Note that every link is within at least one Hyperdoc, just as any primedia is deemed to be within a Hyperdoc. Links are Primedia.


COMPARISON POINTERS
should
CROSS THE
BOUNDARIES
OF FRAMES
Transclusion
implements
THE SAME THING
IN MANY CONTEXTS
C1
C2
C3

COMPARISON POINTERS



should
CROSS THE
BOUNDARIES
OF FRAMES.

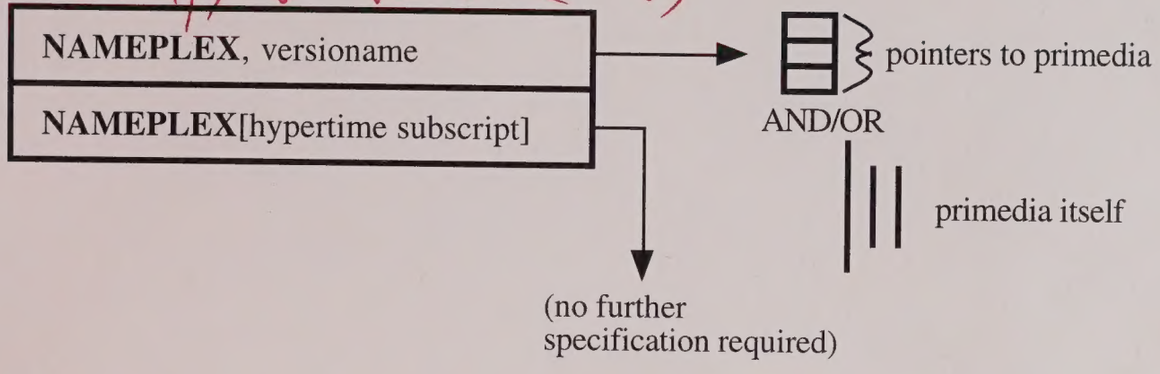


Transclusion
implements
THE SAME THING
IN MANY CONTEXTS.

OSMIC

Preliminary Specifications

More formally, a version :)



[see p. 8, bottom]